Commodore Plus/4 and C16: Hardware and Advanced Basic Programming.

Version 1.01

by Janne Peräaho & Anders Persson

- Introduction 2
- Slow and Fast 2
- Restore key 2
- Window Command 2

- Screen Codes / TED Text modes 3
- Default Character Set 4
- Your own Character Set 6

- Escape codes 6

- ROM 7
- How to access rom programs? 7
- Kernal jump table 7
- 3-plus-1 8

AUTHORS

Janne Peräaho E-mail: amity@surfeu.fi

Pyrytie 11 A 9
90630 Oulu
Finland

Anders Persson http://listen.to/boray

Thanks to Brian Flood for the Plus/4 and to Janne for the cooperation.

ALL DOCUMENTS

This document is part of a document package for Plus/4, C16 and C116 users. The documents included are:

- "basic35.pdf", Janne's original Basic 3.5 manual (more or less)
- "short35.pdf", a Basic Quick Guide.
- "advanced.pdf", This document - The Hardware and Advanced Basic programming.
- "tedmon.pdf", a short description of the built in machine language monitor.

REFERENCES

Commodore Vic-20 and C64 programmers reference manual, Talking to TED by Levente Harsfalvi, Commodore plus/4 and c16 memory map.

INTRODUCTION

The TED chip is the graphics and sound chip of these computers. For a good description and a memory map of it, read this article:

http://www.canberra.edu.au/~scott/C=Hacking/C-Hacking12/gfx.html

Complete Memory maps of the Plus4/C16:

http://www.funet.fi/pub/cbm/maps/C16.MemoryMap
http://plus4.emucamp.com/tools/rommap/index.php

SLOW/FAST

These computers run at two processor speeds (~1.7MHz) and a slow (~0.9MHz) at the same time. Well, not at the same time, but it constantly switches between the fast and slow processor speeds. It's slow whenever the raster beam is on the visible screen, and fast otherwise. The resulting speed is about 1.1MHz for NTSC and 1.2MHz for PAL. If you switch off the display, it will run on the fast speed all the time. You can use this from Basic, just as the FAST and SLOW commands on the C128:

POKE 65286,0 (the screen goes blank) and back: POKE 65286,27

RESTORE KEY?

If you have used a vic20/c64/c128, then you might wonder where the "restore" key is? But there is none! To perform something like it, hold down run/stop and press the reset button. This enters the monitor. Type X and hit return to return to basic. The basic program remains in memory.

I don't know about you, but I find that procedure a little akward. So why not make your own restore key instead? When programming Basic, the most common reason to use the vic20/c64/c128s restore key is to restore the colors, sound and graphics mode to the default. But this can be done with SYS commands! So let's define a function key:

```
KEY 1,"{clr home}SYS65418:SYS65409:SYS65412"+CHR$(13)
```

Then to perform the equavilence to the c64's runstop-restore, first press run/stop to break the program (if it's running), release and then press F1. How nice! If you like, you can put LIST in there too, because that is probably the command you most often like to do after the restore:

```
KEY 1,"{clr home}SYS65418:SYS65409:SYS65412:LIST"+CHR$(13)
```

If you like, you can also put this line in the program you are working on so that you don't need to type it in every time you turn on the computer.

The sys commands used are RESTOR, CINT and IOINIT from the Kernal.

Another little tip (if you don't have made your own restore key) and you just want to get out of a basic graphics mode without having to type GRAPHIC 0. Then just press any key followed by return. The basic will then answer with ?SYNTAX ERROR and go to GRAPHIC 0 by itself to display the error message.

WINDOW COMMAND

The C128 has a WINDOW basic command to set the output window to just a part of the screen. The C16/Plus4 has the same feature but no specific command for it. You can do it with Esc codes (Look at page 6 for the whole list) or with pokes. The Esc codes to use are Esc-T to set top of window, Esc-B to set bottom and Esc-N for the full screen (and clearing it). The pokes to use are 2021 (Screen bottom), 2022 (Screen top), 2023 (Screen left) and 2024 (Screen right). Also, the CHAR command restores the window to the whole screen without clearing it (compared to Esc-N that also clears the screen). If you only use the pokes, then you need to follow up with a PRINT"{home}"; or the output could start outside of the new window. So for something like the C128's window command, you could use this line (which probably is the shortest way of doing it):

```
CHAR, X1,Y1,CHR$(27)+"T":POKE2024,X2:POKE2021,Y2
```

Where (X1,Y1) is the top position and (X2,Y2) is the bottom position.

SCREEN CODES (TED text modes)

The Screen memory starts at **3072** by default.

The color/attribute memory starts at **2048** by default.

Normal Text (the default) (8x8 pixle characters)

- Screen memory: Bits 0-6 is one of 128 characters and bit 7 is reverse on/off.
- Color memory: Bits 0-3: Color, Bits 4-6: luminance, Bit 7: Flash on/off

```
Examples: Poke 3072,1 (puts an A at top left screen position)
          Poke 3073,2+128 (puts a reverted B beside it)
          Poke 2048,128+2+5*16 (makes the A light red and blinking)
```

256 Char display (8x8 pixle characters)

Turn on with Poke 65287,peek(65287) OR 128

- Screen memory: Bits 0-7 is one of 256 characters.
- Color memory: Bits 0-3: Color, Bits 4-6: luminance, Bit 7: Flash on/off

Multi Color (4x8 pixle characters)

With multicolor (lo-res) turned on, you have to design your own characters where every byte is built of four bit pairs = 4 pixles. Every pixle is in memory two bits, so there are four different combinations, 00, 01, 10, and 11 which each represent a different color.

Turn on with poke 65287,peek(65287)or16

- Screen memory: Bits 0-7 is one of 256 characters.
- Color memory:

```
Bit 3: Multicolor on/off. (Both multicolor (4x8) characters and
       hires (8x8) characters can be displayed on the same screen.)
Bits 0-2 Color and
bits 4-6 luminance for the "11"-pixles in the charterset.
```

For the rest of the combinations:

"00"=background (color 0) (address 65301 bits 0-3 color and 4-6 luminance), "01"=color 3 (address 65302) and "10"=(address 65303).

Extended Color (8x8 pixle characters)

(This means you can have different background colors to different hires characters on the same screen)

Turn on with Poke 65286, peek(65286)or64

- Screen memory:

  Bits 0-5 is one of 64 characters.

  Bits 6-7 is background color for the character:

  "00"= normal background (color 0) (65301 bits 0-3 color and 4-6 luminance), "01"= color 3 (65302), "10"= (65303) and "11"= (65304).

- Color memory:

```
Bit 3: Extended color on/off.
Bits 0-2 Foreground Color and
bits 4-6 luminance.
```

(There are also two bitmap modes: Hires (320x200) and Multicolor (160x200). These are easiest used from basic with the GRAPHIC command. The only text modes that works with "split screen" seems to be Normal Text and 256 char display.)

DEFAULT CHARACTER SET:

	SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE	
128	@		0	U	u 149	21	*		42	170
129	A	a	1	V	v 150	22	+		43	171
130	B	b	2	W	w 151	23	,		44	172
131	C	c	3	X	x 152	24	—		45	173
132	D	d	4	Y	y 153	25	.		46	174
133	E	e	5	Z	z 154	26	/		47	175
134	F	f	6	[(Ä)	ä 155	27	Ø		48	176
135	G	g	7	£(Ö)	ö 156	28	1		49	177
136	H	h	8	](Å)	å 157	29	2		50	178
137	I	i	9	↑	158	30	3		51	179
138	J	j	10	←	159	31	4		52	180
139	K	k	11	SPACE	160	32	5		53	181
140	L	l	12	!	161	33	6		54	182
141	M	m	13	"	162	34	7		55	183
142	N	n	14	#	163	35	8		56	184
143	O	o	15	$	164	36	9		57	185
144	P	p	16	%	165	37	:		58	186
145	Q	q	17	&	166	38	;		59	187
146	R	r	18	'	167	39	<		60	188
147	S	s	19	(	168	40	=		61	189
148	T	t	20	)	169	41	>		62	190

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
191 ?		63		T 212	84			106 234
192		64		U 213	85			107 235
193 ♠	A	65		V 214	86			108 236
194	B	66		W 215	87			109 237
195	C	67	♣	X 216	88			110 238
196	D	68		Y 217	89			111 239
197	E	69	♦	Z 218	90			112 240
198	F	70		Ä 219	91			113 241
199	G	71		Ö 220	92			114 242
200	H	72		Å 221	93			115 243
201	I	73	π	222	94			116 244
202	J	74		223	95			117 245
203	K	75	SPACE	224	96			118 246
204	L	76		225	97			119 247
205	M	77		226	98			120 248
206	N	78		227	99			121 249
207	O	79		228	100		✓	122 250
208	P	80		229	101			123 251
209 ●	Q	81		230	102			124 252
210	R	82		231	103			125 253
211 ♥	S	83		232	104			126 254
				233	105			127 255

YOUR OWN CHARACTER SET

First you have to tell that characters should be fetch from RAM. This is done by clearing the second bit of the TED register number 65298 ($FF12):

```
poke 65298, peek (65298) and 251
```

Then you must select a "memory page" where your character set is stored. Each page is 1024 bytes long, so on the C16 there are 16 possible locations:

```
Page 0: characterset starts at memory address 0
Page 1: characterset starts at memory address 1024
Page 2: characterset starts at memory address 2048
...
Page 14: characterset starts at memory address 14336
Page 15: characterset starts at memory address 15360
```

(On the Plus4 there are more pages, probably 64)

Bits 2-7 of the TED register number 65299 ($FF13) determine, which page should be used. To select a page do as follows:

```
poke 65299, (peek (65299) and 3) or (page * 4)
```

Another way of doing the same thing is to take the first byte in the address (as long as one of the pages are used). For example, page 15's address is 15360. The hex number for this is $3c00. So you can just poke 65299, dec("3c").

When using Basic one should protect the character set so, that Basic's program text does not overwrite your character data. This can be done by setting the upper limit for the Basic text, i.e. the highest address Basic is alloved to use. To do that, use memory addresses 55 ($37) and 56 ($38). 55 is the lower half of the 16 bit address and 56 is the upper half.

To switch back to ROM character set, just set the 2nd bit of the 65298 and set the page back to page 52:

```
poke 65298, peek (65298) or 4 : poke 65299,208
```

ESCAPE CODES

Cancel quote and insert mode	ESC O
Cancel started Esc code	ESC X
Erase to end of current line	ESC Q
Erase to start of current line	ESC P
Move to start of current line	ESC J
Move to end of current line	ESC K
Enable auto-insert mode	ESC A
Disable auto insert mode	ESC C
Delete current line	ESC D
Insert line	ESC I
Enable scrolling	ESC L
Disable scrolling	ESC M
Scroll up	ESC V
Scroll down	ESC W
Set bottom of screen window	ESC B
Set top of screen window	ESC T
Set window to full screen minus 1 and clear	ESC R
Set window to full screen and clear screen	ESC N

Example: Press Esc followed by A for auto insert mode, or PRINT CHR$(27);"W" to scroll the screen down from basic.

ROM

The C-16's ROM consists of three main parts: BASIC, TEDMON and KERNAL. BASIC is a Basic interpreter, which can run Basic programs stored in RAM. TEDMON is a machine language monitor for manipulating memory and writing short machine language programs. KERNAL is not a single runnable program, but a collection of system routines arranged as a Jump Table. The Plus/4 also includes a built in software package called "3-plus-1" that includes word processing, spread sheet, business graphics and a database.

HOW TO ACCESS ROM PROGRAMS?

BASIC is the primary user interface and it starts automatically when power is turned on. The secondary user interface is TEDMON, which can be entered by holding down the Reset button when turning the computer on.

You can also activate TEDMON from BASIC and vice versa. See Basic's MONITOR command and TEDMON's X command.

The KERNAL is used by the Basic interpreter and other programs. Using KERNAL routines requires a little machine language knowledge since it is required to use processor registers for exchanging information between a KERNAL routine and the routine caller. It is however possible to call KERNAL routines directly from Basic with the use of the SYS command and the following memory positions: 2034=A, 2035=X, 2036=Y. For example, POKE 2034,23 and the accumulator will contain the value 23 when the SYS command is started.

To call a KERNAL routine do as follows:

1. Set parameters.
2. Call KERNAL routine using JSR command (or SYS from Basic).
3. Handle any return error.

Note, that some routines need to be initialized with other routines before they can be used. For example, using the SAVE routine requires initialization with the SETLFS and SETNAM routines.

See the "KERNAL Jump Table" below for more information on available KERNAL routines.

Basic example:

```
10 SYS DEC("FFE4") : REM "GETIN"
20 A=PEEK(2034) : IF A=0 THEN 10
30 A$ = CHR$(A)
```

Equals to:

```
10 GETKEY A$
```

KERNAL JUMP TABLE

Seems to be exactly the same as for the C64. Please consult the C64 programmer's reference manual for parameters. Available on http://www.funet.fi/pub/cbm/c64/manuals/

Routine	Address	Function
CINT	$FF81	Initializes screen editor.
IOINIT	$FF84	Initializes I/O devices.
RAMTAS	$FF87	Tests RAM.
RESTOR	$FF8A	Restores vectors to initial state.
VECTOR	$FF8D	Changes vectors for user.
SETMSG	$FF90	Controls O.S. messages.
SECOND	$FF93	Sends SA after LISTEN.
TKSA	$FF96	Sends SA after TALK.
MEMTOP	$FF99	Sets/reads top of memory.
MEMBOT	$FF9C	Sets/reads bottom of memory.
SCNKEY	$FF9F	Scans keyboard.
SETTMO	$FFA2	Sets timeout in DMA disk.
ACPTR	$FFA5	Handshakes serial bus or DMA disk byte in.
CIoUT	$FFA8	Handshakes serial bus or DMA disk byte out.
UNTLK	$FFAB	Sends UNTALK out serial bus or DMA disk.
UNLSN	$FFAE	Sends UNLISTEN out serial bus or DMA disk.

LISTN	$FFB1	Sends LISTEN out serial bus or DMA disk.
TALK	$FFB4	Sends TALK out serial bus or DMA disk.
READSS	$FFB7	Returns I/O STATUS byte.
SETLFS	$EF8A	Sets LA, FA, SA.
SETNAM	$FFBD	Sets file name.
OPEN	$FFC0	Opens logical file.
CLOSE	$FFC3	Closes logical file.
CHKIN	$FFC6	Opens channel in.
CHOUT	$FEC9	Opens channel out.
CLRCH	$FECC	Closes I/O channels.
SETLFS	$FFBA	Sets logical file number and command of device.
CHRIN	$FFCF	Inputs one character from keyboard.
CHROUT	$FFD2	Outputs one character to screen.
LOAD	$FFD5	Loads/verifies memory from device.
SAVE	$FFD8	Saves memory area to device.
SETTIM	$FFDB	Sets internal clock.
RDTIM	$FFDE	Reads internal clock.
STOP	$FFE1	Scans STOP key.
GETIN	$FFE4	Gets one character from keyboard queue.
CLALL	$FFE7	Closes all files.
UDTIM	$FFEA	Increments clock.
SCRORG	$FFED	Screen org.
PLOT	$FFF0	Gets/sets cursor position.
IOBASE	$FFF3	Returns location of start of I/O.

3-PLUS-1

The extra software package built into the Plus/4 is started by pressing the F1 key followed by return. The word processor is started by default. For the other parts, hold down Commodore and press C, then type TC (followed by return) for Spreadsheet, TF for file manager (database) or TW to jump back to the word processor.

THE WORD PROCESSOR

Here are the word processor's commands: Press Commodore+C followed by

CA - Directory	CB - Create block	CM - Clear memory
CP - Clear pointers	CT - Clear tabs	DB - Delete block
DF - Delete file	DL - Delete line of text	EP - Erase pointer
IB - Insert block	ID - Format disk	IL - insert text line
LF - Load file	MF - Merge file	PR - Print document
RE - Search and replace	SF - Save file	SP - Set pointer
SR - Search	*P - Prind doc in memory	

Keys:

CLR - move to bottom	Ctrl-= - Set a Tab	Ctrl-9 - reverse (formatting)
Ctrl-0 reverse off	Del - Backspace	C=-C - Command mode
C=-R - move to col 41	C=-L move to left margin	C=-Q - Repeat prev keystroke
C=-@ - Undo return	Home - Home	Instert - Instert
Return - Terminate line	Sh-Return - to left marg.	Shift-= - Tab key

Document formatting: The following keywords can be embedded in the text. They are written in reverse video and typed in lowercase. ":" is used to separate multiple instructions on the same line and ";" is used to terminate the instruction. For example papersize50;

asc	Send ascii character to the printer.
center	Center this line.
justify	Justify text
linkfile	Link documents at print time (with PR). Example: linkfile "opa";
lmarg	Set left marginal. Example: lmarg9; (default is 0)
nextpage	Page break
nojustify	Turn off justify (default)
nowrap	Turns of word wrap (for spreadsheets)
no#page	Deactivates #page command
other	Used with non-Commodore printers (normal Ascii-printers).
pagelen	Sets page length. (Default is 60).
pagepause	Pause printing after every page.
papersize	Changes size of the paper from default 66. Use together with pagelen.
pause	Pause printing until you press return.
rmarg	Set right margin (default is 77).
set#pg	Set page number. Used together with #page.
#page	Print page number at the bottom of the page.
wrapon	Turn word wrap on (default).